Stata's Assert Command

An IR Programmer's Secret Weapon

Ryan Johnson, Ph.D.

UNIVERSITY OF CALIFORNIA, RIVERSIDE

Has this ever happened to you?

- Someone calls you to complain that
  - your new fall enrollment report is missing the new department in the science college
  - your new entering student report shows a huge, unexpected drop in average SAT since last year
  - your new employee report counts way too many support staff
  - your new 5-year enrollment forecast shows a bizarre pattern for several majors
  - your new graduate student report erroneously double counts some students

In every case, the cause is the same:

Unexpected  changes!

Databases are fluid and unpredictable

- New values enter and old values disappear
  - Example: major codes
- Numerical ranges change
  - Example: new GRE scoring schema
- Fields in one location stop being updated and are replaced by fields in others
  - Example: New columns for new SAT test
- Even the meaning of a row can change
  - Example: one row per person becomes multiple rows per person

Ways to handle unexpected changes

- Constantly re-check data assumptions every time you run existing scripts
  - Accurate but takes a lot of time
- Assume nothing will change and fix things only once it's obvious they're wrong
  - Somewhat efficient, but inevitably incorrect results cause problems for campus
- Use automated checks that guarantee key assumptions are correct and avoid errors
  - Highly efficient and accurate

Stata has a command for this

- "Assert" followed by some condition tells Stata to check whether that condition is true.
  - Examples:
    - assert 1==1 will evaluate as true
    - assert 1==0 will evaluate as false
  - If a standard assert evaluates as false, Stata will crash, giving the programmer a chance to identify the problem
  - Programmers can also tell Stata to take any action (e.g., fix problem automatically) rather than just crash if assert fails

Example 1: Checking Strings

```
odbc load, exec("
    select distinct major1
    from sara.irc_students_trd
    where term=201740
") lowercase clear;
```

Load data into Stata's memory using SQL

```
gen mcheck=0;
```

Create new column to check

```
foreach m in ANTH MATH PSYC STAT {;
    replace mcheck=1 if major1=="`m'";
};
```

Loop to define known values

```
assert mcheck==1;
```

Check new column against some logical standard

Example 1: Checking Strings

Step 1

major1
ANTH
MATH
PBPL
PSYC
STAT

Step 2

major1	mcheck
ANTH	0
MATH	0
PBPL	0
PSYC	0
STAT	0

Step 3

major1	mcheck
ANTH	1
MATH	1
PBPL	0
PSYC	1
STAT	1

Step 4

Assert checks every value in this column, determines that this one is <>1, and crashes

Example 1: Checking Strings

```
. gen mcheck=0;

.          foreach m in ANTH MATH PSYC STAT {;
  2.          replace mcheck=1 if major1=="`m'";
  3. };
(1 real change made)
(1 real change made)
(1 real change made)
(1 real change made)

.          assert mcheck==1;
1 contradiction in 5 observations
assertion is false
r(9);

end of do-file
```

Stata stops running code once this happens. This behavior alerts you to the problem and gives you the opportunity to fix it and re-run the script

Example 2: Checking Ranges

```
odbc load, exec("
     select pidm, sattotal
     from sara.irc_students_trd
     where term=201740
") lowercase clear dsn(ppir01);
```

```
assert inrange(sattotal,600,2400) | sattotal==.;
```

You don't have to create a new column. Stata can handle the assert command in memory.

Example 2: Checking Ranges

- The “capture” command records information about the result of the assert in a local variable called “_rc” and allows the script to keep running
- By using _rc, you can tell Stata to take actions based on the result of the assert

Example 2: Checking Ranges

Tells Stata not to crash if assert fails. Records _rc error code.

```
capture assert inrange(sattotal,600,2400) | sattotal==.;

if _rc==0 {;
    display "SAT Total is fine";
};

else {;
    display "Students with inaccurate SAT scores:";
    list pidm sattotal if !inrange(sattotal,600,2400) & sattotal!=.;
    replace sattotal=. if !inrange(sattotal,600,2400);
};
```

Useful for creating exception reports

Useful for fixing data problems you only need to fix in your current results and not in the database itself

Example 2: Checking Ranges

When using syntax that makes a clean output file:

Students with inaccurate SAT scores:

pidm	sattotal
355608	230000
864725	0

Example 3: Checking Proportions

- Sometimes a problematic data change is detected not through values and ranges but through proportions of groups.
- It's helpful to have a data-evaluation script that runs on fresh data and compares proportions of key variables to some standard.
- Example: first-generation status

Example 3: Checking Proportions

```
odbc load, exec("
    select firstgen, count(*) hc
    from sara.irc_students_trd
    where term=201740 and lev1='U'
    group by firstgen
") lowercase clear dsn(ppir01);

egen totaln=total(n);

assert n/totaln >= .50 if firstgen=="Y";
```

Assume anything under 50% is worthy of further inspection

Example 3: Checking Proportions

Step 1

firstgen	n
	32
N	8578
Y	11459

Step 2

firstgen	n	totaln
	32	20069
N	8578	20069
Y	11459	20069

Notice how our method does not account for problems caused by missing data. That would be another good assert to add

Step 3

Assert checks whether n/totaln>=50%.

Example 4: Forecast Modeling

- Complex modeling scripts can use assert to check for certain conditions and alter the models accordingly.
- This removes the need for a human to check assumptions for each new forecast as new data enter the stream.
- Example: forecasting continuation rates by major and level

Example 4: Forecast Modeling

```
logistic enr c.terms i.res;

predict penr;

capture assert penr~=.;

if _rc>0 {;
    summarize enr if penr==.;
    replace penr=r(mean) if penr==.;
};

assert penr~=.;
```

The final forecasts will only make sense if every row in this column has a predicted value

Fill in gaps with average for people missing estimates.

Double check that every row has an estimate

Example 4: Forecast Modeling

Step 1

enr	terms	res	penr
1	1	1	0.89
0	5	0	0.23
0	6		
1	3	1	0.75
...	...	...	...

Step 2

Assert looks for missing data and finds some

Step 3

Stata calculates mean of those with missing penr

Step 4

enr	terms	res	penr
1	1	1	0.89
0	5	0	0.23
0	6		0.30
1	3	1	0.75
...	...	...	...

Stata fills in missing estimates based on previously calculated mean

Example 5: Database Updates

- If you maintain your own database, you can use assert to ensure new uploads do not violate any important conditions
  - Examples:
    - One row per person
    - A particular map of majors to colleges
    - Each student flagged as new must really be new
    - No missing grades
    - Course instructors must be current employees
    - Students paid financial aid were enrolled

Example 5: Database Updates

```
import delimited newtermdata.csv, clear;

bysort pidm: assert _n==1;

odbc insert, table("students_trd");
```

A simple Stata trick for checking uniqueness of rows by some set of columns

Example 6: Ad Hoc Requests

- Programming custom requests is faster and more accurate when you can test assumptions while you work without having to examine them one by one
  - Examples:
    - Every row in one data file has a match in another
    - The sum of instructor teaching responsibilities is always 100%
    - 5-year grad rate ≤ 6-year grad rate
    - Staff growth rates are always positive
    - Every tenured faculty member has a valid home department

Example 6: Ad Hoc Requests

```
use datafile1, clear;

merge 1:1 pidm using datafile2;

assert _merge!=1;

drop if _merge==2;
```

Joins the two files on pidm and creates column called _merge showing different types of matches

Prove no rows in datafile1 were unmatched

Get rid of rows in datafile2 that are not in datafile1

Example 6: Ad Hoc Requests

Datafile1

pidm
1
2
3
4

Datafile2

pidm
1
2
3
5
6

Both files merged

pidm	_merge
1	3
2	3
3	3
4	1
5	2
6	2

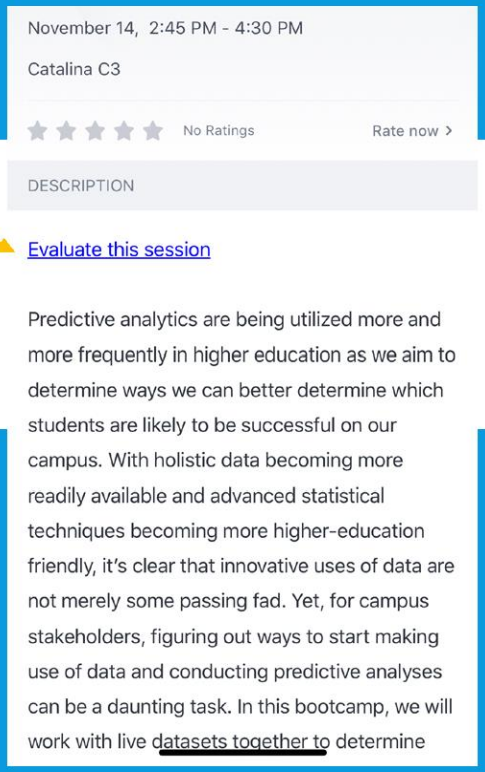
Use assert to prove you are only getting the _merge results you expect

Questions?

ryan.johnson@ucr.edu

PLEASE FILL OUT AN EVALUATION FOR THIS SESSION

CLICK on the EVALUATION LINK IN YOUR CAIR APP



THANK YOU!